COMPARING CONFIGURATION INFORMATION FOR A DATA FORWARDING DEVICE

§ 1. BACKGROUND OF THE INVENTION

§ 1.1 FIELD OF THE INVENTION

The present invention concerns configuring data forwarding devices, such as routers for example. More specifically, the present invention concerns comparing candidate (or other) configuration information for such a device with previously stored (e.g., committed) configuration information.

§ 1.2 RELATED ART

The description of art in this section is not, and should not be interpreted to be, an admission that such art is prior art to the present invention. The present invention may be used for comparing configuration information for data forwarding devices.

Data forwarding devices, such as routers and switches, may be interconnected to form networks. The interconnections may be such that each data forwarding device has a plurality of input lines and a plurality of output lines. A basic function of these devices is to forward data received at their input lines to the appropriate output lines. Routers, for example, may determine the appropriate output lines based on a destination address(es) contained in the received data and forwarding tables. Switches may be configured so that data received at input lines are transferred out appropriate output lines.

Such data forwarding devices may need to be configured appropriately. This may be done by entering configuration commands and/or information through a keyboard or other type of interface into a data forwarding

device. Other types of information or commands may also be entered into the device through the keyboard or interface.

When entering a command or information through a keyboard (or similar user interface) of a data forwarding device or editing an existing configuration, there is a risk that the command or information may be entered or edited incorrectly or incompletely. Such errors may cause the data forwarding device to be incorrectly configured and may lead to serious malfunctions.

Further, as can be appreciated from the foregoing, configuring data forwarding devices, such as routers for example, can be a complex task, often requiring networking expertise. Further, configuration information can become quite large, thereby increasing the likelihood of mistakes in configuration.

Accordingly, there is a need to reduce the risk that commands or information will be entered incorrectly or incompletely and to ensure that the edits are entered correctly.

§ 2. SUMMARY OF THE INVENTION

The disclosed invention helps users to detect errors in (candidate) configuration information by permitting the (candidate) configuration information to be compared with previously saved configuration information. Differences between the two sets of configuration information may be indicated by special characters or symbols preceding changed lines of configuration, or by special font characteristics (e.g., color, underlining, typeface, font size, font type, etc.) applied to changed versus unchanged lines or sections of the configuration.

The disclosed invention may also operate on configuration information relevant to data forwarding devices, such as routers for example. Further, some configuration information may include instructions and parameters.

The disclosed invention may operate to compare only instructions, only parameters, or both.

Each of the sets of configuration information may include configuration categories. Such categories may include chassis configuration information, class-of-service configuration information, firewall configuration information, forwarding-options configuration information, groups configuration information, interfaces configuration information, policy-options configuration information, protocols configuration information, routing-instances configuration information, routing-options configuration information, network management protocol configuration information, and/or system configuration information.

For such hierarchical configuration information, the comparison may be limited to particular hierarchical levels and categories. For example, the part to be compared may be a particular hierarchical level of a particular category, and its descendant statements. If one of the sets is a candidate set of configuration information being worked on by a user, then the particular hierarchical level and particular category defining parts to be compared may correspond to that being worked on the by user. Alternatively, the user may define the hierarchical levels and/or categories to be compared.

Finally, different hierarchical levels and categories may have different permission requirements. In this way, the users permitted to access and/or edit various hierarchical levels and categories of configuration information may be limited.

§ 3. BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a high-level bubble chart diagram of a simple exemplary data forwarding device having configuration information on which the present invention may operate.

Figure 2 is a high-level bubble chart diagram of an exemplary data forwarding device having configuration information on which the present invention may operate.

Figure 3 is a high-level bubble chart diagram of exemplary configuration operations that may take place in the device of Figure 1, or the device of Figure 2.

Figure 4 is a high-level flow diagram of an exemplary method that may be used to effect at least some of the exemplary configuration operations of Figure 3.

Figure 5 is a high-level flow diagram of an exemplary method that may be used to effect a configuration compare operation in accordance with the present invention.

Figure 6 is a block diagram that illustrates different types of configurations that may be used in a router configuration data structure.

Figure 7 illustrates an exemplary configuration hierarchy in a protocols configuration part of a set of router configuration information.

Figure 8 illustrates exemplary instructions and parameters for a chassis part of a set of router configuration information.

Figure 9 illustrates exemplary instructions and parameters for a class-of-service part of a set of router configuration information.

Figure 10 illustrates exemplary instructions and parameters for a firewall part of a set of router configuration information.

Figure 11 illustrates exemplary instructions and parameters for a forwarding options part of a set of router configuration information.

Figure 12 illustrates exemplary instructions and parameters for a groups part of a set of router configuration information.

Figures 13a-13d illustrate exemplary instructions and parameters for an interfaces part of a set of router configuration information.

Figure 14 illustrates exemplary instructions and parameters for a policy options part of a set of router configuration information.

Figures 15a-15k illustrate exemplary instructions and parameters for a protocols part of a set of router configuration information.

Figure 16a-16c illustrate exemplary instructions and parameters for a routing-instances part of a set of router configuration information.

Figures 17a-17c illustrate exemplary instructions and parameters for a routing-options part of a set of router configuration information.

Figure 18 illustrates exemplary instructions and parameters for a simple network management protocol part of a set of router configuration information.

Figures 19a and 19b illustrate exemplary instructions and parameters for a system part of a set of router configuration information.

Figure 20 is a high-level block diagram that illustrates an exemplary machine that may be used to effect various operations of the present invention.

§ 4. DETAILED DESCRIPTION

The present invention involves novel methods, apparatus and data structures for comparing at least a part of sets of configuration information, such as configuration information of a router, or hierarchical configuration information. The following description is presented to enable one skilled in the art to make and use the invention, and is provided in the context of particular applications and their requirements. Various modifications to the disclosed embodiments will be apparent to those skilled in the art, and the general principles set forth below may be applied to other embodiments and applications. Thus, the present invention is not intended to be limited to the embodiments shown and the inventor regards his invention as the following disclosed methods, apparatus and data structures and any other patentable subject matter.

§ 4.1 ENVIRONMENTS IN WHICH THE PRESENT INVENTION MAY OPERATE

Various aspects of the present invention may use configuration information, such as that used by data forwarding devices (e.g., routers). Two exemplary data forwarding devices are described below.

§ 4.1.1 FIRST EXEMPLARY DATA FORWARDING DEVICE

Figure 1 is a high-level bubble chart diagram of a simple exemplary data forwarding device 100 having configuration information 150 on which the present invention may operate. As shown, a data forwarding operation 110 may use information in a forwarding table 120 to forward incoming data (e.g., packets) towards a final destination. The forwarding table 120 may be generated and updated by a path-to-forwarding information translation operation 130. The path-to-forwarding information translation operation 130 may perform its generation and update functions based on a path (e.g., routing) table 140 and

device configuration information 152. The path (e.g., routing) table 140 may be generated by a path (e.g., route) determination operation 160 based on network state (e.g., link state) information, as well as device configuration information 152. For example, the path determination operation 160 may operate in accordance with known routing protocols to populate a routing table.

A control instruction user interface operation 170 may be used for, among other things, generating, importing, and/or editing the device configuration information 152. The present invention may concern at least a part of the control instruction user interface operation 170.

§ 4.1.2 SECOND EXEMPLARY DATA FORWARDING DEVICE

Figure 2 is a high-level bubble chart diagram of an exemplary data forwarding device 200 having configuration information 150' on which the present invention may operate. The data forwarding device 200 may include a data (e.g., packet) forwarding facility 210 and a path (e.g., route) determination facility 260. Basically, the data forwarding facility 210 may function to forward data towards its ultimate destination, and the path determination facility 260 may function to generate and/or update a forwarding table 120' based on path (e.g., route) determinations.

In an exemplary embodiment, the data forwarding facility 210 may include an operating system (micro) kernel 220 which supports various operations (e.g., an interface operation 230 and a chassis operation 250). The exemplary data forwarding facility 210 may also include an instance of a forwarding table 120b' used to forward data towards its destination. The forwarding table instance 120b' may correspond to an instance of the forwarding table 120a' of the path determination facility 260.

In an exemplary embodiment, the path determination facility 260 may include an operating system kernel 262 which supports various operations (e.g., a path (e.g., route) determination operation 264, an interface operation 266, a chassis operation 268, control instruction user interface operations 170', etc.) and which may be used to generate the forwarding table instance 120a'. The path (e.g., route) determination operation 264 may be used to determine a path (e.g., routing) table 140'. Network management (e.g., SNMP) operations 270 may interact with the various operations 264,266,268 supported by the operating system kernel 262. The control instruction user interface operation 170' may act on configuration information 150' in accordance with the present invention.

As shown in the blow-up of bubble 170' in Figure 2, the control instruction user interface operations 170' may include user login and authentication operations 271, configuration operations 272 and control instruction editing operations 276 (which may access recently executed control instructions stored in the buffer 277). As will be appreciated from the description of the present invention, the configuration operation 272 may operate on present candidate configuration information 273.

In both the device 100 of Figure 1 and the device 200 of Figure 2, control instruction user interface operations 170 and 170', respectively, may interact with device configuration information 150 and 150', respectively. The present invention may constitute a part of such control user interface operations 170/170'. The present invention may be used with other data forwarding devices.

§ 4.2 FUNCTIONS THAT MAY BE PERFORMED BY THE PRESENT INVENTION

A data forwarding device (referred to below as a "router", without loss of generality) may be provided with default configuration information to permit its components to interoperate properly. Users may want to change such

default configuration information, for example, to optimize the router for a particular intended use, to have the router operate properly within a given network, to use the router with various physical communications media, etc.

A routing protocol operation may be used to control the routing protocols that run on a router. Such an operation may start all configured routing protocols and handle all routing messages. This operation may maintain one or more routing tables, which may consolidate, into common tables, routing information learned from a plurality of protocols. A user may configure the router to control the routes that a protocol places into each table and the routes from that table that the protocol advertises. This may be done be defining one or more routing policies and then applying such policies to the specific routing protocol.

From this routing information, the routing protocol operation may determine the active routes to network destinations and may install these routes into a forwarding table. Finally, the routing protocol operation may implement a routing policy, which a user may use to control routing information transferred between routing protocols and the routing table (i.e., information may be filtered (e.g., using firewalls) so that only some of it is transferred, and properties associated with routes may be set).

A user may also configure and control physical interface devices and logical interfaces in a router. For example, the user may configure various interface properties such as the interface location, interface encapsulation, and interface-specific properties.

Further, a user may configure and control chassis-related properties of a router, such as conditions that trigger alarms and clock sources.

The present invention may function to help users to detect errors in (candidate) configuration information by permitting the (candidate) configuration information (e.g., of data forwarding devices such as routers) to be compared with a previously saved configuration information. Differences between at least a part of the two sets of configuration information may be indicated by special characters or symbols preceding changed lines of configuration information, or by special font characteristics (e.g., color, underlining, typeface, font size, font type, etc.) applied to changed versus unchanged lines (or "statements") or sections of the configuration information.

The present invention may further help users to detect errors in candidate configuration information by applying "scoping" functionality, so that in the context of a hierarchical configuration, only relevant or selected parts of a set of candidate configuration information and a previously saved set of configuration information are compared. Different hierarchical levels and/or categories of configuration information may be stored as different objects in an object-oriented database. The present invention may limit permissions to view or change different hierarchical levels and/or categories of configuration information.

Finally, the invention may further help users to detect errors in (candidate) configuration information by comparing only instructions, only parameters, or both.

§ 4.3 EXEMPLARY OPERATIONS, ARCHITECTURE, METHODS AND DATA STRUCTURES

Operations that may be performed by the present invention, exemplary methods and data structures that may be used to effect such operations, and exemplary hardware that may be used to effect such operations are described below.

§ 4.3.1 EXEMPLARY OPERATIONS

Figure 3 is a high-level bubble chart diagram of exemplary configuration operations 272' that may take place in a data forwarding device, such as the device of Figure 1, or the device of Figure 2. Notice that the configuration operations 272' may include a plurality of particular configuration operations 310. Such particular configuration operations 310 may include, for example, an ACTIVATE operation for removing an inactive tag from an instruction, an ANNOTATE operation for annotating an instruction with a comment, a COMMIT operation for committing to current candidate configuration information, a COPY operation for copying an instruction, a DEACTIVATE operation for adding an inactive tag to an instruction, a DELETE operation for deleting a data element, an EDIT operation for editing a sub-element, an EXIT operation for leaving a current hierarchical level of configuration information, a HELP operation for providing help information, an INSERT operation for inserting a new ordered data element, a LOAD operation for loading configuration information (e.g., from an ASCII file), a QUIT operation for quitting a hierarchical level of configuration information, a RENAME operation for renaming an instruction, a ROLLBACK operation for rolling back a database to a selected one of previously committed versions of configuration information, a RUN operation for running an instruction, a SAVE command for saving configuration information, a SET operation for setting a parameter, a SHOW operation for showing a parameter, a STATUS operation for displaying a user status, a TOP operation for navigating to a top hierarchical level of configuration information, and an UP operation for navigating to a next higher hierarchical level of configuration information.

As indicated by Figure 3, the configuration operations 272' may further include a COMPARE CONFIGURATION operation 310c. Briefly stated, the COMPARE CONFIGURATION operation 310c may accept at least a part of a default set of configuration information (e.g., the last committed configuration

information 152') or at least a part of a selected one of other past sets of configuration information 154' as a first input and at least a part of a set of present candidate (or other past) configuration information 273' as a second input. The COMPARE CONFIGURATION operation 310c will then generate changes 320 to the part(s) of the set of default or selected configuration information 154'/152' needed to get the corresponding part(s) of the set of (present candidate) configuration information 273'. Alternatively, the COMPARE CONFIGURATION operation may operate on at least a part of any two sets of configuration information.

If a user is in a given hierarchical level of a given category within the candidate configuration information 273', then the COMPARE CONFIGURATION operation 310c may limit its comparison to the relevant hierarchical level and its descendants, of the given category within the set of default or selected configuration information 154'/152'. Alternatively, a user may select categories and/or hierarchical levels (i.e., part(s)) of the sets of configuration information on which the COMPARE CONFIGURATION operation 310c may operate.

The changes to the default/selected configuration information may be visually indicated by special symbols preceding instructions added, removed, or changed, by special font characteristics, etc.

§ 4.3.2 EXEMPLARY METHODS AND DATA STRUCTURES

Figure 4 is a high level flow diagram of an exemplary method 272" that may be used to effect at least some of the exemplary configuration operations 310 of Figure 3. First, as indicated at conditional branch point 410, it is determined whether or not a (logged in) user has proper permission to perform any configuration operations. If not, access is denied and such denial may be flagged as indicated by optional block 415, before the method 272" is left via

RETURN node 490. If, on the other hand, it is determined that the user has proper permission to perform any (i.e., at least some) configuration operations, the method 272" continues to step 420 where an empty set of candidate configuration information is opened. Then, as indicated by block 425, the user may, assuming they have the appropriate level of permission, load preexisting configuration information (Recall, e.g., the LOAD and ROLLBACK operations.), view and navigate through the candidate configuration (Recall, e.g., the EDIT, EXIT, QUIT, SHOW, TOP, and UP operations.), change instructions and/or parameters (Recall, e.g., the ACTIVATE, DEACTIVATE, DELETE, INSERT, SET, and RENAME operations.), add instructions and/or parameters (Recall, e.g., the COPY and INSERT operations.), test instructions and/or parameters (Recall, e.g., the RUN operation), and change or add comments to instructions (Recall, e.g., the ANNOTATE operation).

During such a session, the candidate configuration may be saved as indicated by decision branch point 440. In one exemplary embodiment, candidate configuration information is not used until it is committed. As shown by optional decision branch point 450, if the candidate configuration information is committed, it may be checked for proper syntax as indicated by block 460. If there are any syntactical errors, the user may be notified of such errors as indicated by decision branch point 470 and block 475, before the method 272" branches back to block 425. If, on the other hand, there are no syntactical errors, the candidate configuration information may be activated and marked or flagged as the current active configuration information, as indicated by decision branch point 470 and block 480, before the method 272" is left via RETURN node 490.

Recall from Figure 3 and block 425 of Figure 4 that a number of operations are made available during a configuration session. One such operation may be a "compare configuration information operation" 310c. Before describing an exemplary method for effecting a compare configuration

information operation with reference to Figure 5, exemplary data structures for storing configuration information are first described below. An appreciation of an exemplary hierarchical data structure for configuration information will be useful in understanding optional "scoping" and "permissions" aspects of the present invention.

As stated above, the set of configuration information may have, or be arranged in the context of, a hierarchy. In one exemplary configuration information data structure, the configuration information is defined by a hierarchy of statements. In this exemplary data structure, there are two types of statements -- container statements and leaf statements. Container statements contain other statements, while leaf statements do not contain other statements. All of the container and leaf statements collectively define the configuration hierarchy. In this exemplary embodiment, each statement at the top level of the configuration hierarchy resides at the trunk (or root) level of a tree data structure. These top-level statements are often container statements that contain other statements that form branches of the tree data structure. The leaf statements form the leaves of the tree data structure. An individual hierarchy of statements (that starts at the trunk) may be referred to as a "statement path." Various statements and their inter-relationships may be stored as objects in an object-oriented database.

Such a hierarchical data structure may be used for storing configuration information for a data forwarding device, such as a router for example. Figure 6 illustrates exemplary container statements contained in an exemplary configuration information structure 152" for a router. As indicated, the highest level of the hierarchy may include a number of configuration categories, such as chassis configuration 605, class-of-service configuration 610, firewall configuration 615, forwarding options configuration 620, groups configuration 625, interfaces configuration 630, policy-options configuration 635, protocols configuration 640, routing instances configuration 645, routing options

configuration 650, simple network management protocol (SNMP) configuration 655 and system configuration 660.

Figure 8 illustrates exemplary statements and parameters for a chassis configuration part 605 of an exemplary router configuration 152". Figure 9 illustrates exemplary statements and parameters for a class-of-service configuration part 610 of an exemplary router configuration 152". Figure 10 illustrates exemplary statements and parameters for a firewall configuration part 615 of an exemplary router configuration 152". Figure 11 illustrates exemplary statements and parameters for a forwarding options configuration part 620 of an exemplary router configuration 152". Figure 12 illustrates exemplary statements and parameters for a groups configuration part 625 of an exemplary router configuration 152". Figure 13, which includes Figures 13a through 13d, illustrates exemplary statements and parameters for an interfaces configuration part 630 of an exemplary router configuration 152". Figure 14 illustrates exemplary statements and parameters for a policy options configuration part 635 of an exemplary router configuration 152". Figure 15, which includes Figures 15a through 15k, illustrates exemplary statements and parameters for a protocols configuration part 640 of an exemplary router configuration 152". Figure 16, which includes Figures 16a through 16c, illustrates exemplary statements and parameters for a routing-instances configuration part 645 of an exemplary router configuration 152". Figure 17, which includes Figures 17a through 17c, illustrates exemplary statements and parameters for a routing-options configuration part 650 of an exemplary router configuration 152". Figure 18 illustrates exemplary statements and parameters for a simple network management protocol configuration part 655 of a router configuration 152". Finally, Figure 19 illustrates exemplary statements and parameters for a system configuration part 660 of an exemplary router configuration 152".

Referring now to the exemplary instructions and parameters of Figure 15 for a protocols configuration part 640 of an exemplary router configuration 152", the following statement path:

```
protocols{
    ospf{
        area 0.0.0.0{
            interface so-0/0/0{
                hello interval 5;
            }
            interface so-0/0/1{
                hello interval 5;
            }
        }
    }
}
```

is illustrated by the tree 640' of Figure 7.

In this example, the "protocols" statement is a top-level statement at the trunk of the configuration tree. The "ospf", "area", and "interface" statements are all subordinate container statements of a higher statement in the configuration tree (i.e., the define branches). In this case, each of the "interface" statements contain a parameter value (so-0/0/0 and so-0/0/1). Finally, the "hello interval" statement is a leaf on the configuration tree. In this case, each of the "hello interval" statements contain a parameter value (5) as the length of the hello interval, in seconds, for each of the defined interfaces.

This statement path is depicted with cross-hatching and bold lines in Figure 7. As illustrated, the "hello interval" leaf statement, at a fifth hierarchical level 740, is contained in the "interface" branch container statement. The "interface" branch container statement at the fourth hierarchical level 730 is, in turn, contained in the "area" branch container statement. The "area" branch container statement at the third hierarchical level 720 is, in turn, contained in the "ospf" branch container statement. Finally, the "ospf" branch container statement

at the second hierarchical level 710 is, in turn, contained in the "protocols" root container statement at the first hierarchical level.

In the configuration statements set forth above, the hierarchical levels are defined within an open brace symbol "{" and a closed brace symbol "}". If a statement at a given hierarchical level is empty (i.e., if it contains no other statement), then the braces need not be depicted. Finally, each leaf statement (or the non-leaf statement at the otherwise lowest level of the hierarchy) may be depicted with a semicolon.

Having illustrated examples of hierarchical data structures for configuration information, an exemplary method 310c' that may be used to effect the compare configurations operation 310c is now presented with reference to Figure 5. As indicated by block 510, at least a part of a set of (present candidate) (router) configuration information is accepted. As indicated by block 530, at least a part of a set of default (e.g., most recently committed) configuration information may also be accepted. Alternatively, in one embodiment as indicated by optional conditional branch point 520 and optional block 540, a user may choose (at least a part of) one of a plurality of sets of particular configuration information (e.g., one of nine stored previously committed). Thus, at least a part of two sets of configuration information are accepted at this point. In another alternative (not shown), a user can select (at least a part of) two sets of previously committed or stored configuration information.

Referring next to optional block 550, the hierarchical level of the set of candidate configuration information which the user is presently within may be determined. This may be used to scope the compare operation to the present hierarchical level and category, and it's descendants. Then, as indicated by decision branch point 560, it is determined whether or not the user has permission to perform the compare operation (at the given hierarchical level or

category). If not, the method 310c' may be left via RETURN node 590. Otherwise, if the user has proper permission(s), the method 310c' continues to block 570 where the part(s) of the set of the present candidate (or other) configuration information is compared with the part(s) of the set of default or chosen committed (router) configuration information. This comparison may be effected using the UNIX "diff" command, a similar technique, or any other known comparison technique. Note that user permission may have been previously checked, in which case decision branch point 560 is redundant.

Still referring to block 570, in one embodiment, comparisons may be made as follows. First, a copy of the candidate configuration may be made. All further operations may use this "scratch" copy of the candidate configuration. Each object (e.g., hierarchical level) of the configuration may be associated with two flags: a "referenced" flag; and a "created" flag. The configuration to which the candidate (or other) configuration is to be compared may be loaded into a configuration database. As each object is loaded into the database, its "referenced" flag (and its parents, in the configuration object hierarchy) is set (e.g., to "1"). When an object's data value is set in the database, it is determined whether or not a value currently exists for that object. If so, the "created" flag is set (e.g., to "1"), and the old value is stored before being replaced with a new value. After the configuration is loaded, the selected part of the database is exported (e.g., displayed) in ASCII. As the selected part is being displayed, the two flag values are checked and are processed as indicated in the following table:

"created" flag	"referenced" flag	Meaning	Display
0	0	Object is new in candidate configuration	Display "+" symbol
0	1	Object is unchanged in candidate configuration	No special display
1	0	Object is deleted in candidate configuration	Display "-" symbol
1	1	Object has new value in candidate configuration	Display "+" symbol for new value and "-" symbol for old value

The "scratch" copy of the candidate configuration can be discarded after the comparison.

In the alternative embodiment in which optional step 550 is performed, the comparison may be limited to the determined present hierarchical level and category, and any of its descendants. For example, referring to Figure 7, if the user was in the "interface" category of the fourth hierarchical level 730, only the interface configuration information and all configuration information descending from (i.e., contained in) the interface category would be compared. This illustrates default scoping. In a further alternative embodiment, at block 550, the user could enter an explicit information for scoping the comparison. For example, referring to Figure 7, the user could specify that the comparison be only for the "area" category of the third hierarchical level 720, and all configuration information descending from the "area" category (only some of which is shown in Figure 7). In this way, selected part(s) of, rather than the entire, sets of configuration information may be compared.

Regardless of what is compared, the comparison results may be printed, displayed, and/or saved as indicated block 580, before the method 310c' is left via RETURN node 590. For example, configuration information removed in

the candidate configuration information may be denoted by a "-" symbol, and additions may be denoted by a "+" symbol. Other symbols, or font attributes may be used to denote additions, deletions and changes.

§ 4.3.3 EXEMPLARY HARDWARE ARCHITECTURES

Figure 20 is high-level block diagram of a machine 2000 which may effect one or more of the operations, and store one or more of the data structures, discussed above. The machine 2000 basically includes a processor(s) 2010, an input/output interface unit(s) 2030, a storage device(s) 2020, and a system bus(es) and/or a network(s) 2040 for facilitating the communication of information among the coupled elements. An input device(s) 2032 and an output device(s) 2034 may be coupled with the input/output interface(s) 2030. Operations of the present invention may be effected by the processor(s) 2010 executing instructions. The instructions may be stored in the storage device(s) 2020 and/or received via the input/output interface(s) 2030. The instructions may be functionally grouped into processing modules.

The machine 2000 may be a router for example. In an exemplary router, the processor(s) 2010 may include a microprocessor and/or (e.g., custom) integrated circuit(s). In the exemplary router, the storage device(s) 2020 may include ROM, RAM, SDRAM, SRAM, SSRAM, DRAM, flash drive(s), hard disk drive(s), and/or flash cards. At least some of these storage device(s) 2020 may include program instructions defining an operating system, a protocol daemon, and/or other daemons. In a preferred embodiment, the methods of the present invention may be effected by a microprocessor executing stored program instructions (e.g., defining a part of the protocol daemon). At least a portion of the machine executable instructions may be stored (temporarily or more permanently) on the storage device(s) 2020 and/or may be received from an external source via an input interface unit 2030.Finally, in the exemplary router, the input/output interface unit(s) 2030, input device(s) 2032 and output device(s)

2034 may include interfaces to terminate communications links. The input device(s) 2032 may include a keyboard.

Naturally, the operations of the present invention may be effected on systems other than routers. Such other systems may employ different hardware and/or software.

§ 4.4 OPERATIONAL EXAMPLE IN AN EXEMPLARY EMBODIMENT

The following example illustrates an operation of an exemplary compare configuration method. Suppose that the selected or last committed router configuration information is:

```
...
    protocols{
        ospf{
            area 0.0.0.0{
                interface so-0/0/0{
                    hello interval 5;
                    transmit interval 30;
                }
                interface so-0/0/1{
                    hello interval 5;
                    transmit interval 30
                }
            }
        }
    }
...
```

where the ellipses denote configuration information preceding or following the printed protocols configuration information. Suppose further that the candidate router configuration information is:

...

```
        protocols{
            ospf{
                area 0.0.0.0{
                    interface so-0/0/0{
                        hello interval 10;
                        priority 1;
                    }
                    interface so-0/0/1{
                        hello interval 10;
                        priority 3
                    }
                }
            }
        }
...
```

Assuming that the compare configuration operation is scoped to protocols, ospf, area, the output would be:

```
...
        protocols{
            ospf{
                area 0.0.0.0{
                    interface so-0/0/0{
-                       hello interval 5;
+                       hello interval 10;
-                       transmit interval 30;
+                       priority 1;
                    }
                    interface so-0/0/1{
-                       hello interval 5;
+                       hello interval 10;
-                       transmit interval 30;
+                       priority 3
                    }
                }
            }
        }
...
```

Notice that since the parameter values of the "hello interval" statements were changed from 5 to 10 in the candidate router configuration

information, these changes are denoted with "-" and "+" symbols preceding such statements. Notice also that since the "transmit interval" configuration information are not found in the candidate router configuration information, these changes are denoted with "-" symbols. Finally, notice that since the "priority" configuration information is added in the candidate router configuration information, this added information is denoted with "+" symbols.

A control instruction user interface may permit results of earlier control instructions (e.g., SHOW) to be "piped" through a compare instruction. The following control instruction could be used to compare a (e.g., a present hierarchical level and its descendants of) candidate configuration information with (e.g., that of) a second most recently committed set of configuration information:

show | compare rollback 2

where the "|" symbol denotes a piping operation.

§ 4.5 CONCLUSIONS

As can be appreciated from the foregoing disclosure, the present invention helps users to detect errors in a candidate configuration information, for example, before committing to that candidate configuration information. Optional scoping capabilities may be used to limit a compare configurations operation, thereby permitting users to work on smaller, more manageable parts of sets of configuration information. By providing an optional permissions check, only authorized users can create new configuration information and compare it to previously committed configuration information. Such permissions may limit certain users to certain classes of configuration information and/or certain hierarchical levels of the configuration information. Thus, for example, a user authorized to work on "protocols", "routing-instances" and "routing options"

configuration information may be prevented from working on “chassis” and “interfaces” configuration information.